Hough Transform

Mohamed EL ANSARI

Visiting Scholar
EMARO, Ecole Centrale de Nantes, Nantes Cedex, France

Associate Professor
Department of Computer Science, Faculty of Sciences,
University of Ibn Zohr
Agadir, Morocco

m.elansari@univ-ibnzoehr.ac.ma
melansari@gmail.com
Content

- Introduction
- Hough transform for straight lines
- Hough transform for circles
- Hough transform for analytic curves
Performed after edge detection.

Introduced to detect curves from images.

Classical Hough transform used to detect straight lines, circles, ellipses and parabolas.
  - Analytic curves

Generalized Hough Transform used to find arbitrarily complex shapes.
  - Non-analytic curves (Not covered here)

Map a difficult *pattern detection problem* into a simple *peak detection problem* in space of the parameters of the curve.
Voting scheme

- Let each feature vote for all the models that are compatible with it.
- Hopefully the noise features will not vote consistently for any single model.
- Missing data doesn't matter as long as there are enough features remaining to agree on a good model.
Hough Transform

An early type of voting scheme.

General outline:

- Discretize parameter space into bins.
- For each feature point in the image, put a vote in every bin in the parameter space that could have generate this point.
- Find bins that have the most votes.

A line (straight) can be represented as:
\[ y = mx + b \]
A line in image space can be represented by point in Hough space:
What does a point \((x_0, y_0)\) in the image space map to in the Hough space?

- Answer: the solution of \(b = -x_0 m + y_0\)
- This is a line in Hough space.
What does a point \((x_0, y_0)\) in the image space map to in the Hough space?

- **Answer:** the solution of \(b = -x_0m + y_0\)
- **This is a line in Hough space.**

![Diagram showing the mapping of a point from image space to Hough parameter space.](image-url)
Where is the line that contains both \((x_0, y_0)\) and \((x_1, y_1)\)?

- It is the intersection of the lines \(b = -x_0 m + y_0\) and \(b = -x_1 m + y_1\).
Where is the line that contains both \((x_0, y_0)\) and \((x_1, y_1)\)?

- it is the intersection of the lines \(b = -x_0 m + y_0\) and \(b = -x_1 m + y_1\)

\[
\begin{align*}
\text{Image space} \\
\begin{array}{c}
y \\
y_1 \\
y_0 \\
x_0 & x_1 \\
x 
\end{array}
\quad\quad\quad
\begin{array}{c}
\text{Hough parameter space} \\
b \\
b = -x_0 m + y_0 \\
b = -x_1 m + y_1 \\
m
\end{array}
\end{align*}
\]
Hough transform for straight lines detection

slope-intercept

Mohamed EL ANSARI (EMARO-ECN)

Hough Transform
angle-radius

- Problems with the \((m, b)\) space:
  - Unbounded parameter domain.
  - Vertical lines require infinite \(m\).
- Alternative solution: polar representation
Angle-radius

- Problems with the \((m, b)\) space:
  - Unbounded parameter domain.
  - Vertical lines require infinite \(m\).
- Alternative solution: polar representation
angle-radius

- Problems with the \((m, b)\) space:
  - Unbounded parameter domain.
  - Vertical lines require infinite \(m\).

- Alternative solution: polar representation
Problems with the \((m, b)\) space:
- Unbounded parameter domain.
- Vertical lines require infinite \(m\).

Alternative solution: **polar representation**

\[ x \cos \theta + y \sin \theta = \rho \]
Angle-radius

- Each point from the image space will add a sinusoid in the \( \rho - \theta \) space.

\[
\rho = x \cos \theta + y \sin \theta
\]

- A straight line in image space will be represented by a point in parameter space \((\rho - \theta)\), the intersection of the sinusoids.
Algorithm outline

Hough Transform Algorithm

- Initialize all the accumulator H cells to zero
- For each edge point \((x, y)\) in the image
  - For \(\theta = 0\) to 180
    - \(\rho = x\cos\theta + y\sin\theta\)
    - \(H(\theta, \rho) = H(\theta, \rho) + 1\)
  - end
- end
- Find the value(s) of \((\theta, \rho)\) where \(H(\theta, \rho)\) is a local maximum
  - The detected line in the image is given by
    - \(\rho = x\cos\theta + y\sin\theta\)
Basic illustration

Features

Votes
Other shapes

Square

Circle
Several lines
More complicated images

http://ostatic.com/files/images/ss_hough.jpg
Effect of noise

- Peaks get fuzzy and hard to locate.
Effect of noise

- Peaks get fuzzy and hard to locate.
Effect of noise

- Number of votes for a line with 20 points with increasing noise:
Effect of noise

Random points

- Uniform noise can lead to spurious peaks in the array (accumulator)
Effect of noise

Random points

- As the level of uniform noise increases, the maximum number of votes increases too:
Practical details

- Try to get rid of irrelevant features
  - Take only edge points with significant gradient magnitude
- Choose a good grid / discretization
  - Too coarse: large votes obtained when too many different lines correspond to a single bucket
  - Too fine: miss lines because some points that are not exactly collinear cast votes for different buckets
- Increment neighboring bins (smoothing in accumulator array)
Incorporating image gradients

- The gradient direction of edge points can be used during votes.
- Computational cost of the algorithm can be reduced

Modified Hough Transform

For each edge point \((x, y)\)

- \(\theta = \text{gradient orientation at } (x, y)\)
- \(\rho = x \cos \theta + y \sin \theta\)
- \(H(\theta, \rho) = H(\theta, \rho) + 1\)

\[
\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \, \frac{\partial f}{\partial y} \end{bmatrix} \\
\]

\[
\theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right)
\]
Pros of Hough Transform

- Can deal with occlusion.
- Can detect multiple instances of a model in a single pass.
- Some robustness to noise.
- Tolerant of gaps in the edges.
Cons of Hough Transform

- Complexity of search time increases exponentially with the number of model parameters.
- It’s hard to pick a good grid size.
- Non-target shapes can produce spurious peaks in parameter space.
Hough transform of circles

- How many dimensions will the parameter space have?
- Given an oriented edge point, what are all possible bins that it can vote for?
Parameter space representation

- In Cartesian coordinates, the equation of a circle is given by:

\[(x - a)^2 + (y - b)^2 = r^2\]

where \((a, b)\) is the center of the circle and \(r\) its radius.
- Parameter space → \(a\), \(b\) and \(r\).
- For a pixel (edge) belonging to circle, what is the locus for the parameters of that circle?
Parameter space representation

- In Cartesian coordinates, the equation of a circle is given by:
  \[(x - a)^2 + (y - b)^2 = r^2\]
  
  where \((a, b)\) is the center of the circle and \(r\) its radius.
- Parameter space \(\rightarrow a, b\) and \(r\).
- For a pixel (edge) belonging to circle, what is the locus for the parameters of that circle?

Right circular cone
In Cartesian coordinates, the equation of a circle is given by:

\[(x - a)^2 + (y - b)^2 = r^2\]

where \((a, b)\) is the center of the circle and \(r\) its radius.

A point in image space provides a right circular cone in Hough space.

The right circular cones corresponding to the edge points belonging to a circle will intersect in a point \((a_0, b_0, r_0)\) (the parameters of the circle).
In Cartesian coordinates, the equation of a circle is given by:

\[(x - a)^2 + (y - b)^2 = r^2\]

where \((a, b)\) is the center of the circle and \(r\) its radius.

A point in image space provide a right circular cone in Hough space.

The right circular cones corresponding to the edge points belonging to circle will intersect in in a point \((a_0, b_0, r_0)\) (the parameters of the circle)
In Cartesian coordinates, the equation of a circle is given by:

\[(x - a)^2 + (y - b)^2 = r^2\]

where \((a, b)\) is the center of the circle and \(r\) its radius.

A point in image space provide a right circular cone in Hough space.

The right circular cones corresponding to the edge points belonging to circle will intersect in in a point \((a_0, b_0, r_0)\) (the parameters of the circle)
Let us denote the equation of a circle as $f(x, a) = 0$, where $x = (x \ y)^T$ is an image point and $a = (a \ b \ r)^T$ is a parameter vector.

**Hough algorithm**

1. Form an array $A(a)$ (accumulator array), initially set to zero.
2. For each edge pixel
   1. Compute all $a$ such that $f(x, a) = 0$
   2. Increment the corresponding accumulator array entries $A(a) = A(a) + 1$
3. Local maxima in the array $A$ correspond to curves (circles) in the image.
Using directional information

- Gradient information reduces one more free parameter.
- The center of the circle must lie $r$ units along the direction of the gradient.
- $\frac{df}{dx}(x, a) = 0$
- $\frac{dy}{dx} = \tan[\phi(x) - \frac{\pi}{2}]$ where $\phi(x)$ is the gradient direction
- The parameter locus is reduced to a line:

![Diagram of parameter locus reduced to a line](image-url)
Using directional information

- Gradient information reduces one more free parameter.
- The center of the circle must lie $r$ units along the direction of the gradient.
- \( \frac{df}{dx}(x, a) = 0 \)
- \( \frac{dy}{dx} = \tan[\phi(x) - \frac{\pi}{2}] \) where $\phi(x)$ is the gradient direction
- The parameter locus is reduced to a line:
Using directional information

- Gradient information reduces one more free parameter.
- The center of the circle must lie $r$ units along the direction of the gradient.
- $\frac{df}{dx}(x, a) = 0$
- $\frac{dy}{dx} = \tan[\phi(x) - \frac{\pi}{2}]$ where $\phi(x)$ is the gradient direction
- The parameter locus is reduced to a line:
Using directional information

- Gradient information reduces one more free parameter.
- The center of the circle must lie $r$ units along the direction of the gradient.
- $\frac{df}{dx}(x, a) = 0$
- $\frac{dy}{dx} = \tan[\phi(x) - \frac{\pi}{2}]$ where $\phi(x)$ is the gradient direction
- The parameter locus is reduced to a line:
Using directional information

- Gradient information reduces one more free parameter.
- The center of the circle must lie $r$ units along the direction of the gradient.
- $\frac{df}{dx}(x, a) = 0$
- $\frac{dy}{dx} = \tan[\phi(x) - \frac{\pi}{2}]$ where $\phi(x)$ is the gradient direction.
- The parameter locus is reduced to a line:

![Diagram showing the parameter locus as a line in a 3D coordinate system with axes $a$, $b$, and $r$. The point $(x, y)$ is marked on the graph.](image)
Using directional information

Modified Hough algorithm

1. Form an array $A(a)$ (accumulator array), initially set to zero.
2. For each edge pixel
   1. Compute all $a$ such that $f(x, a) = 0$ and $\frac{df}{dx}(x, a) = 0$
   2. Increment the corresponding accumulator array entries $A(a) = A(a) + 1$
3. Local maxima in the array $A$ correspond to curves (circles) in the image.
Modified Hough algorithm

1. Form an array $A(a)$ (accumulator array), initially set to zero.
2. For each edge pixel
   1. Compute all $a$ such that $f(x, a) = 0$ and $\frac{d}{dx}(x, a) = 0$
   2. Increment the corresponding accumulator array entries $A(a) = A(a) + 1$
3. Local maxima in the array $A$ correspond to curves (circles) in the image.
Cost of computation

If we have $m$ parameters each have $M$ values

- Without directional information
  - The computation is proportional to $M^{m-1}$

- With directional information
  - The computation is proportional to $M^{m-2}$
Hough transform for analytic curves

Let us denote the equation of a curve as $f(x, a) = 0$, where $x = (x \ y)^T$ is an image point and $a$ is a parameter vector (Hough space).

Hough algorithm for analytic curve

1. Form an array $A(a)$ (accumulator array), initially set to zero.
2. For each edge pixel
   1. Compute all $a$ such that $f(x, a) = 0$
   2. Increment the corresponding accumulator array entries $A(a) = A(a) + 1$
3. Local maxima in the array $A$ correspond to curves in the image.
S. Lazebnik, S. Seitz, S. Savarese, J. Hays, D. Hoiem, and Others
*Slide Credits.*

D. H. Ballard  
*Generalizing the Hough Transform to Detect Arbitrary Shapes*  

R. O. Duda and P. E. Hart  
*Use of the Hough Transformation to Detect Lines and Curves in Pictures*  